May 1984 LIDS-P-1376

WINDOW PROTOCOLS IN PRESENCE OF CAPTURE*

by

Moshe Sidi**

ABSTRACT

The effect of capture on window-type algorithms in a slotted ALOHA broadcasting network is investigated. It is assumed that the nodes of the network are divided into two groups and only packets sent by nodes of one of the groups might be captured. We consider both the situations that the receiver can distinguish between success slots and capture slots and that it cannot. For each of these situations window-type multiple access algorithms are described and their performance in terms of attainable throughputs is evaluated.

---

*This research was carried out at the Massachusetts Institute of Technology, Laboratory for Information and Decision Systems with partial support provided by the National Science Foundation under grant NSF-ECS-8310698.

**Moshe Sidi is with the Department of Electrical Engineering, Technion-Isareal Institute of Technology, Haifa, Israel. He is currently a Post-Doctoral Associate at the Laboratory for Information and Decision Systems, M.I.T., Cambridge, MA 02139, USA.

1. Introduction

In this paper we present and analyze several multiple access protocols suited to time-slotted capture-type channels. These protocols are adapted from the window protocol first introduced by Gallager [1] and subsequently improved by Mosely [2]. The original window protocol [1] was devised for collision-type channels where the underlying assumption is that whenever two or more packets are transmitted during the same slot, then neither of them is correctly received at the common receiver. In fact, this assumption provides a lower bound to the performance of real networks, since in many communication systems, the stronger of two or more overlapping packets may capture the receiver and thus be received without error. As we shall subsequently show, it is possible to improve the performance of the system by taking advantage of this capture effect.

The capture effect has been investigated in [5]-[6] for the classic slotted ALOHA random access scheme, under the non-realistic hypothesis that the combined traffic of new and retransmitted packets forms a stationary and independent Poisson process. Collision resolution algorithms with capture that are adapted from the tree algorithm first introduced by Capetanakis [3] have been presented and analyzed in [4].

The window protocol makes use of a feedback information channel that informs the nodes of the network immediately at the end of each slot of what happened during that slot. In absence of captures, the possible events during a slot are: (i) Idle slot; no packet is transmitted during the slot; (ii) Success slot; exactly one packet is transmitted during the slot and therefore correctly received; (iii) Collision slot; two or more

packets are transmitted during the slot and neither of them is correctly received. When packets might be captured, an additional possible event is: (iv) Capture slot; two or more packets are transmitted during the slot and one of them is correctly received. As in [1], we assume that when events (i), (ii) and (iii) occur the receiver broadcasts through the feedback channel LACK, ACK or NACK respectively. If (iv) occurs we consider two possible situations. In the first we assume that the receiver can distinguish between success slots and capture slots. In the second it cannot. These two situations distinguish between two possible feedback information channels, and we will investigate them separately.

To facilitate the exposition of the protocols and to ease their analysis, our detailed presentation and analysis is confined to the situation that the nodes of the network are divided in advance into two groups, and this division is fixed in time. Only packets transmitted by nodes from one of the groups can be captured at the common receiver. The exact model with its basic assumptions and the description of the two possible feedback information channels are given in Section 2. In Section 3 we review the basic window protocol [1] whose understanding is essential in order to understand the window protocols developed. In Sections 4 and 5 we describe the window protocols for the case that the receiver can distinguish between success and capture slots and for the case that it cannot, respectively. The protocols are evaluated in terms of their maximal throughput in Section 6 and Section 7 contains some extensions to the basic model.

2. The Model

We consider the accessing by a very large, effectively infinite, number of nodes, of a common receiver. The forward channel to the receiver is a noiseless time-slotted capture-type common radio channel. The nodes transmit packets of a fixed length, taken as one slot, and are synchronized in the sense that each transmitted packet arrives at the receiver over exactly one slot. The nodes of the network are divided into two groups, one is called the Dominating Group (DG) and the other the Non-Dominating Group (NDG). Only packets transmitted by nodes from the DG can be captured at the receiver. The nodes of the DG and the NDG generate new packets according to Poisson processes with rates λ_{DG} and λ_{NDG}, respectively.

The possible events that may occur during each slot in a capture-type channel are: (a) Idle slot; no node is transmitting during the slot, in which case the slot is idle; (b) Success slot; exactly one node (from either the DG or the NDG) uses the channel, in which case its packet is successfully received; (c) Collision slot; two or more nodes from the DG with any number of nodes from the NDG or no node from the DG and at least two nodes from the NDG use the slot, in which case individual transmitted packets cannot be reconstructed at the receiver and must be retransmitted at some later time; (d) Capture slot; exactly one node from the DG and one or more nodes from the NDG use the channel during the slot. In this case we assume that the receiver captures the packet transmitted by the node from the DG, therefore it is successfully received at the receiver. All other packets that have been sent by nodes from the NDG during the capture slot cannot be reconstructed at the receiver and must be retrans-

mitted at some later time.

The feedback channel from the common receiver is assumed to be a noiseless broadcast channel. Through this channel the receiver informs all nodes of the network, immediately at the end of each slot, of what happened during that slot. If either of the above events (a), (b) or (c) occurs, then the receiver broadcasts LACK, ACK or NACK respectively. If the above event (d) occurs, i.e. whenever a packet from the DG is captured, we consider two possible feedback information channels.

The first is called feedback with capture (FWC). With FWC the receiver is able to distinguish between capture slots and success slots, i.e. it is able to detect that it has received correctly a packet, though at least one more packet has been simultaneously transmitted. In this case the receiver broadcasts CAPT to all nodes in the network whenever (d) occurs.

The second feedback information channel we consider is called feedback without capture (FWOC). With FWOC the receiver cannot distinguish between capture slots and success slots, i.e. it cannot decide whether a correctly received packet has been the only transmitted packet, or whether other packets have been transmitted at the same time. In this case the receiver broadcasts ACK to all nodes in the network. However, had a capture occurred, such an ACK might confuse the nodes from the NDG that have been transmitting, since they will not be able to know that their packets have not been successfully received. Therefore, with FWOC we assume that each node adds one bit at the head of a transmitted packet. This added bit is 0 or 1 if the node belongs to the DG or to the NDG respectively. Consequently, with FWOC, the receiver broadcasts ACK_0 or ACK_1 when it receives correctly a packet from a node that belongs to the DG or to the NDG respectively. In any case, we assume that propagation delay is negligible, so that the feedback information for a certain slot can be used to determine who should transmit in the next slot.

3. The Basic Window Protocol

An understanding of Gallager's basic window protocol [1], is necessary in order to understand the window protocols for capture-type channels. Therefore we first provide a short description of this protocol.

In the basic window protocol each node of the system keeps track of an interval of time in the past which shall be called a window. The left and right boundaries of the window will be denoted by A and B respectively. During each slot, all packets that arrive during the current window are transmitted. The system may reside in one of three states, S_0, S_1, S_2 depending on what knowledge the nodes have regarding the current window. Remember that here only LACK, ACK or NACK can be received.

In state S_0, the system does not have any knowledge regarding the number of packets that arrived within the current window. For LACK or ACK received while residing in state S_0 the system remains in S_0 and slides the window forward. The new window size will be chosen between μ and T_c-B, whichever is smaller ($A \leftarrow B$, $B \leftarrow B + \min(\mu, T_c - B)$. Here μ is the length of the new window, and is chosen so that some performance measure is optimized, and T_c is the current time. For NACK the system enters state S_2 with the knowlege that the window contains at least two packets. In this transition the window is partitioned into a left part and a right part according to some parameter p (that is again chosen so that some performance measure is optimized) and the left part becomes the new window ($B \leftarrow B_p + A(1-p)$).

If the system is in state S_2 then LACK or NACK maintains the system in state S_2. For NACK the new window is obtained as in the transition from S_0 to S_2. For LACK it is clear that the right part of the previous

window contains at least two packets, therefore it is partitioned and a new window is chosen as in the transition from S_0 to S_2 ($A \leftarrow B$; $B \leftarrow B + (B-A)(1-p)$). An ACK allows the system to transit to state S_1, where the nodes know that the right part of the previous window, which becomes the new window ($A \leftarrow B$; $B \leftarrow [B-A(1-p)]/p$) contains at least one packet.

Finally, from state S_1 an Ack or Nack cause the system to transit to S_0 or S_2 respectively. In the former case the new window is obtained by sliding the window forward in time and expanding it to length μ or T_c-B, whichever is smaller. In the latter case, the window is partitioned and the new window is chosen as in the transition from S_0 to S_2.

4. Window Protocols for Feedback With Capture

In feedback with capture (FWC) the receiver can distinguish between success slots and capture slots. Therefore the possible feedback messages are LACK, ACK, NACK and CAPT. Subsequently we present two window protocols for FWC that are adapted from the basic window protocol described in the previous section. In principal, the difference between the two protocols is that in the first (Protocol 1) at most one packet can be captured between two successive visits to state S_0, while in the second (Protocol 2), the number of captured packets between such two successive visits might be also two. The detailed description of the two protocols appear in Appendix 1.

In both protocols, the nodes of the two groups (the DG and the NDG) keep track of different windows. Let (A_1,B_1) and (A_2,B_2) be the windows of DG nodes and NDG nodes respectively. Basically, for both protocols the nodes perform the basic window protocol for LACK, ACK or NACK using their own parameters, p_1,μ_1 for the DG nodes and p_2,μ_2 for the NDG nodes. For both protocols, when the system is in state S_0 or S_1 and a capture occurs then it is clear that the corresponding window of the NDG nodes contains at least one packet. Therefore, in this case, the DG nodes wait until the window of the NDG nodes involved in the capture will be resolved (a window is said to be resolved whenever the system re-enters state S_0). This is achieved by choosing a zero length window for the DG nodes with no change in the window for the NDG nodes, and transit to state S_1.

The two protocols differ whenever a capture occurs while the system is in state S_2. Note that in this case, it is known that the current window of the NDG nodes contains at least one packet, and that the right part of

the previous nonzero window of the DG nodes also contains at least one packet. A reasonable approach in this case is to resolve the two windows known to contain at least one packet, separately, and then to choose fresh ones. This is what is done in Protocol 1. According to this protocol, after a capture while the system is in state S_2, the NDG window involved in the capture is first resolved, while the DG nodes wait. Then the right part of the previous nonzero window of the DG nodes is resolved while the NDG nodes wait. This is achieved by first choosing a window of zero length for the DG nodes, raising a flag and resolving the window of the NDG. When this window is resolved, a window of zero length is chosen for the NDG, the flag is lowered and the right part of the previous nonzero window of the DG is resolved. Consequently, it is clear that at most one capture can occur between two successive visits to state S_0

Obviously, more information is extracted from a capture than from a simple successful transmission. With this in mind, we expect that a protocol that allows for more than one capture between two successive visits to state S_0, will perform better. This idea is used in Protocol 2. According to this protocol, if a capture occurs while the system is in state S_2, then the NDG window involved in the capture is first resolved while DG nodes wait, as in Protocol 1. After this window is resolved, DG nodes choose their right part of the previous nonzero window as their new window as before but now NDG nodes do not wait but choose a completely fresh window of length μ_2' or T_c-B_2, whichever is smaller and continue the protocol with partitioning parameter p_2'. In this case the system transits to state S_1. If we are lucky and a capture occurs, then the protocol

continues as described above. However, if a collision occurs then it is known that the window of the DG nodes contains at least two packets and no information has been gained regarding the NDG window and it can be considered as fresh. Therefore the protocol continues as before except that the NDG window is reduced to have a zero length.

5. Window Protocols for Feedback Without Capture

With feedback without capture (FWOC) the receiver cannot distinguish between capture and success slots. Therefore it always broadcasts ACK_0 and ACK_1 when receiving correctly a packet from a node from the DG or the NDG respectively. Subsequently, we present two window protocols for FWOC. In principal, in the first protocol (Protocol 3), each time after a non-empty DG window is resolved, DG nodes wait until the NDG nodes will resolve their current window. The motivation is to allow the NDG nodes that were involved in a capture slot to transmit before a new window for the DG nodes is chosen. In the other protocol (Protocol 4) that turns to be, as expected, more efficient, the DG nodes choose a new window immediately after resolving their previous window. The detailed descriptions of the protocols appear in Appendix 1.

In both protocols (Protocols 3 and 4) nodes of the DG and the NDG perform their basic window protocol except for the case that a packet from a DG node is successfully transmitted (i.e. ACK_0 is received). Note that in this case it is possible that the window of the NDG nodes is not empty. When ACK_0 is received while the system is in state S_2, then in both protocols since nothing is known regarding the winodw of the NDG nodes, their window is extended, so it contains also the right part of the previous window. When ACK_0 is received while the system is in either state S_0 or S_1, then in protocol 3 the NDG nodes continue their protocol with no immediate change in the window while the DG nodes wait (by reducing their window to zero length) until the window of the NDG nodes is resolved. It is clear that this is not the best we can do, because in

this case no further statistical knowledge was gained about the current NDG window. Therefore we can improve the performance of the protocol by simply choosing completely new windows for both the DG and the NDG nodes. This is exactly what is done in Protocol 4. Notice that in Protocol 4 packets from NDG nodes might be delayed for long periods if several successive captures occur.

6. Performance Evaluation

In this section, we determine the set of arrival rates λ_{DG} and λ_{NDG} of new packets to nodes of the DG and NDG nodes respectively for which the system is stable. These rates clearly depend on the fresh windows lengths μ_1 and μ_2 and on the partitioning parameters p_1 and p_2 (in Prococol 2 also on μ_2' and p_1'). In particular, we obtain the optimal fresh windows μ_1^* and μ_2^* and partitioning parameters p_i^* and p_2^* so that the total throughput of the system $\lambda_T = \lambda_{DG} + \lambda_{NDG}$ is maximized.

Let us first define some notations (here $\underline{p} = (p_1, p_2)$.

$L_{n,k}^{DG}(\underline{p})$ $\triangleq$ the average number of successfully transmitted packets by nodes of the DG until the next visit of state S_0, given that the current windows contain n and k packets of the DG and NDG nodes, respectively.

$L_{n,k}^{NDG}(\underline{p})$ $\triangleq$ the average number of successfully transmitted packets by nodes of the NDG until the next visit of state S_o, given that the current windows contain n and k packets of the DG and NDG nodes, respectively.

$m_{n,k}(\underline{p})$ $\triangleq$ the average number of slots until the next visit of state S_0, given that the current windows contain n and k packets of DG and NDG nodes, respectively.

$L_{n,k}(\underline{p}) \triangleq L_{n,k}^{DG}(\underline{p}) + L_{n,k}^{NDG}(\underline{p})$

In Appendix 2 we present the equations from which all the above defined quantities can be calculated recursively.

Assuming that the arrival processes of new packets to nodes of the DG and the NDG are Poisson with rages λ_{DG} and λ_{NDG} respectively, we have that the average number of successfully transmitted packets from DG nodes

and NDG transmitted between two successive visits to state S_0 is given by:

$$L^{DG}(\underline{p},\underline{x}) = \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} L^{DG}_{i,j}(\underline{p}) \frac{e^{-(x_1+x_2)} x_1^i x_2^j}{i!\ j!}$$

$$L^{NDG}(\underline{p},\underline{x}) = \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} L^{NDG}_{i,j}(\underline{p}) \frac{e^{-(x_1+x_2)} x_1^i x_2^j}{i!\ j!}$$

where

$$x_1 \triangleq \lambda_{DG}\mu_1 \quad ; \quad x_2 = \lambda_{NDG}\mu_2$$

and

$$\underline{x} \triangleq (x_1, x_2)$$

The total average number of successful transmissions between two successive visits to S_0 is:

$$L(\underline{p},\underline{x}) = \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} L_{i,j}(\underline{p}) \frac{e^{-(x_1+x_2)} x_1^i x_2^j}{i!\ j!}$$

The average number of slots elapsed between two successive visits to S_0 is:

$$m(\underline{x},\underline{p}) = \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} m_{i,j}(\underline{p}) \frac{e^{-(x_1+x_2)} x_1^i x_2^j}{i!\ j!}$$

The total throughput of the system is finally given by:

$$T(\underline{x},\underline{p}) = \frac{L(\underline{x},\underline{p})}{m(\underline{x},\underline{p})}$$

Clearly, the total throughput depends on the partitioning parameters p_1 and p_2 and on the lengths of new windows μ_1 and μ_2, through the quantities x_1 and x_2. We can numerically find the optimal values of p_1^*, p_2^*, x_1^*, x_2^* that maximize the total throughput T and then find the optimal windows μ_1^* and μ_2^*. The values of these parameters along with the corresponding maximal throughputs appear in Table 1.

Protocol	μ_1^*	μ_2^*	p_1^*	p_2^*	λ_{DG}^*	λ_{NDG}^*	max throughput $\lambda_{DG}^*+\lambda_{NDG}^*$	
1	2.51	2.61	0.48	0.49	0.295	0.293	0.588	
2	2.54	2.60	0.48	0.49	0.293	0.296	0.589	$\mu_2^* = 1.52$ $p_2^* = 0.49$
3	2.95	3.17	0.47	0.49	0.190	0.350	0.540	
4	2.10	4.01	0.47	0.49	0.329	0.244	0.573	

Table 1

It is interesting to see that for FWC the maximal throughput is obtained for almost the same arrival rates to DG nodes and NDG nodes and that there is almost no difference between the performance of Protocols 1 and 2. Protocol 3 favors higher arrival rate to NDG nodes since successful transmissions from DG nodes that are not caused by captures lead to wasted slots. Protocol 4 favors higher arrival rates to DG nodes because in FWOC they have strong priority over NDG nodes.

As in the basic window protocol [1] we found that the throughput of the system is not very sensitive to the partitioning parameters p_1 and p_2.

In Figure 1 the region of arrival rates for which the system is stable for $p_1 = p_2 = 0.5$ is below the depicted line for each protocol. There is almost no difference between Protocols 1 and 2. Protocol 3 behaves quite badly for high arrival rates to DG rates since many slots are wasted in that case. Notice that the basic window protocol (without capture) corresponds to $\lambda_{DG} = 0$ or $\lambda_{NDG} = 0$ in fig. 1 (except for Protocol 3). From fig. 1 we can conclude that Protocols 1 and 4 should be used for FWC and FWOC, respectively.

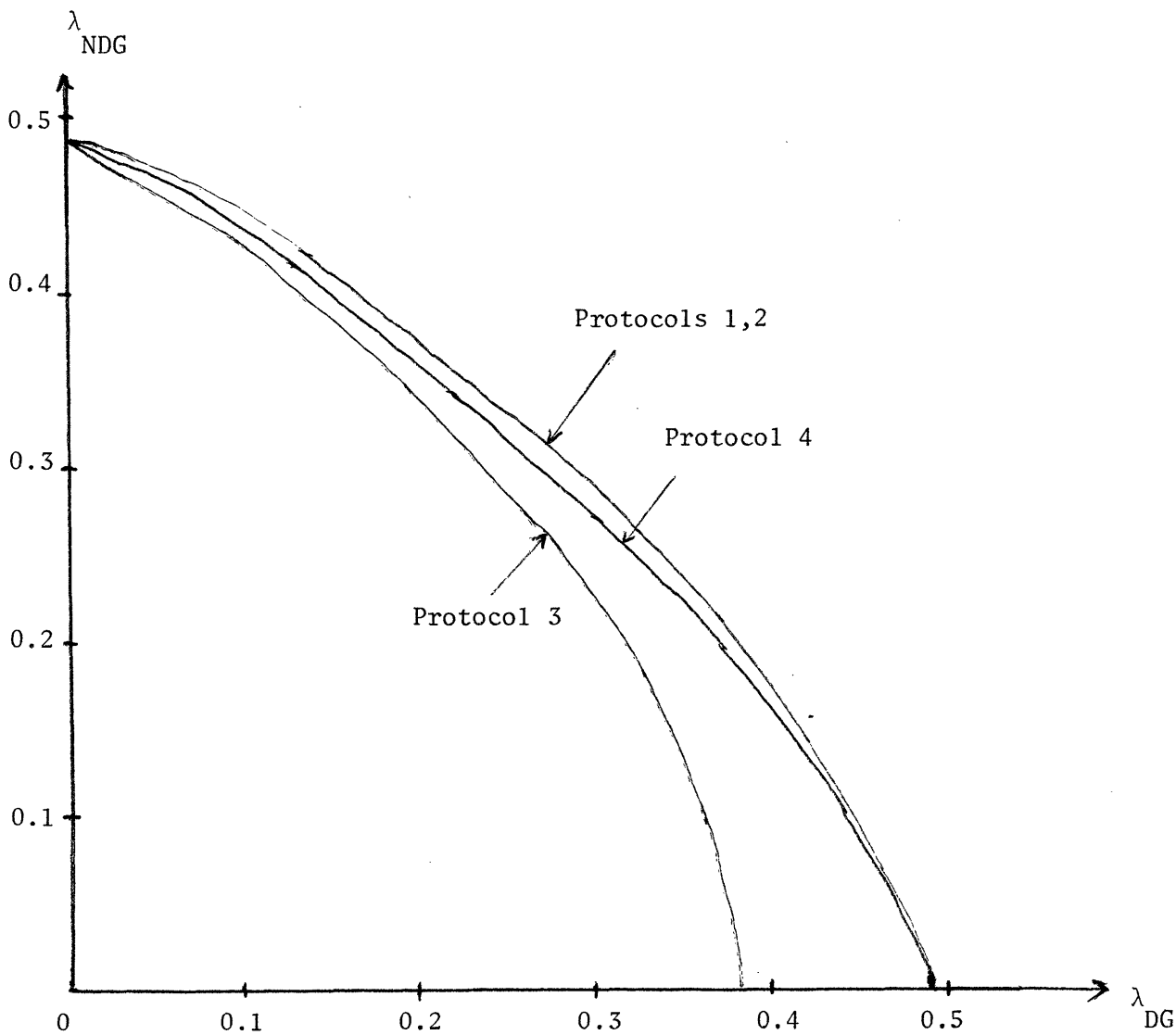

Fig. 1: Regions of Achievable Throughputs

7. Discussion

In this paper we have investigated the effect of capture on the basic window protocol [1], [2]. We have considered both the situations that the receiver can distinguish between success slots and capture slots and that it cannot. For each of these cases we described window-type multiple-access protcols and evaluated their performance. As expected, we have shown that the performance of the system might be significantly improved when packets might be captured.

The exposition of the protocols and their analysis has been confined to the case that the nodes of the system are divided into two groups and only nodes of one of the groups (DG nodes) could be captured. Moreover, we assumed that a capture event does not depend on the number of NDG nodes transmitting along with a node from the DG.

A more realistic case is that a packet transmitted by a node from the DG is captured only if less than K NDG nodes are transmitting at the same time. In some sense K corresponds here to the realative powers of DG and NDG nodes. K=1 corresponds to the case of no capture. The protocols that we presented earlier with very slight modification can be used in this case as well and their performance can be evaluated using the same technique we presented. It turns out, for example, that for FWC using Protocol 1 and K=2, the performance of the system (in terms of attainable throughputs) degraded by about 2% compared with the case of very large (effectively infinite) K. This behavior is due to the fact that the windows chosen by the nodes will very rarely contain large number of nodes. Consequently, as long as $K \geq 2$ the behavior of the system is almost independent of k.

Another possible extension to our basic model is to divide the nodes of the system into N groups instead of only two groups. Let the N groups of the system be denoted by A_1, A_2,...,A_N. We say that group A_i dominates group A_j whenever $i > j$. The domination has the following sense: if a single packet originated at A_i and an arbitrary number of packets originated at groups dominated by A_i, are transmitting during the same slot, then the common receiver will capture the packet originated at A_i

Obviously, in previous sections we considered the case N=2. Similar approach can be applied for $N > 2$. We may note here that for proper operation of the protocols there will be need that each group will have its own identity. It is expected that as N increases the performance of the system will be improved and for very large N the utilization of the channel will be almost perfect. It will be interesting to see what is the rate of improvement as N gets large.

Appendix 1

The parameters A_1, B_1 and A_2, B_2 correspond to the boundaries of the windows for DG nodes and NDG nodes respectively.

Protocol 1

	Nodes of the DG	Nodes of the NDG
S_0:	If there is a packet with arrival time between A_1 and B_1, then transmit it.	If there is a packet with arrival time between A_2 and B_2, then transmit it.
For:	LACK, ACK	LACK, ACK
	$A_1 \leftarrow B_1$	$A_2 \leftarrow B_2$
	$B_1 \leftarrow B_1 + \min(\ _1, T_c - B_1)$	$B_2 \leftarrow B_2 + \min(\ _2 + T_c - B_2)$
	Go to S_0	Go to S_0
For:	CAPT	CAPT
	$A_1 \leftarrow B_1$	
	Go to S_1	Go to S_1
For:	NACK	For NACK
	$B_1 \leftarrow B_1 p_1 + A_1(1-p_1)$	$B_2 \leftarrow B_2 p_2 + A_2(1-p_2)$
	Go to S_2	Go to S_2
S_1:	If there is a packet with arrival time between A_1 and B_1, then transmit it.	If there is a packet with arrival time between A_2 and B_2, then transmit it.
For:	ACK	For: ACK
	If FL=0 then do:	If FL=0 then do:
	$A_1 \leftarrow B_1$	$A_2 \leftarrow B_2$
	$B_1 \leftarrow B_1 + \min(\mu_1, T_c - B_1)$	$B_2 \leftarrow B_2 + \min(\mu_2, T_c - B_2)$
	Go to S_0	Go to S_0

If FL=1 then do:

$B_1 \leftarrow [b_1 - A_1^m(1-p_1)]/p_1$

$FL \leftarrow 0$

Go to S_1

For: CAPT

$A_1 \leftarrow B_1$

G_o to S_1

For: NACK

$B_1 \leftarrow B_1 p_1 + A_1(1-p_1)$

Go to S_2

S_2: If there is a packet with arrival time between A_1 and B_1, then transmit it.

For: LACK

$Temp \leftarrow A_1$

$A_1 \leftarrow B_1$

$B_1 \leftarrow B_1 + (B_1 - Temp)/(1-p_1)$

Go to S_2

For: ACK

$Temp \leftarrow A_1$

$A_1 \leftarrow B_1$

$B_1 \leftarrow [B_1 - Temp(1-p_1)]/p_1$

Go to S_1

For: NACK

$B_1 \leftarrow B_1 p_1 + A_1(1-p_1)$

Go to S_2

If FL=1 then do:

$A_2 \leftarrow B_2$

$FL \leftarrow 0$

Go to S_1

For: CAPT

Go to S_1

For: NACK

$B_2 \leftarrow B_2 p_2 + A_2(1-p_2)$

Go to S_2

If there is a packet with arrival time between A_2 and B_2, then transmit it.

For: LACK

$Temp \leftarrow A_2$

$A_2 \leftarrow B_2$

$B_2 \leftarrow B_2 + (B_2 - Temp)/(1-p_2)$

Go to S_2

For: ACK

$Temp \leftarrow A_2$

$A_2 \leftarrow B_2$

$B_2 \leftarrow [B_2 - Temp(1-p_2)]/p_2$

Go to S_1

For: NACK

$B_2 \leftarrow B_2 p_2 + A_2(1-p_2)$

Go to S_2

For: CAPT

$A_1^m \leftarrow A_1$

$A_1 \leftarrow B_1$

$FL \leftarrow 1$

Go to S_1

For: CAPT

$FL \leftarrow 1$

Go to S_1

Protocol 2

Same as Protocol 1 except for the following changes in state S_1.

	Nodes of the DG	Nodes of the NDG
S_1:	If there is a packet with arrival time between A_1 and B_1, then transmit it.	If there is a packet with arrival time between A_2 and B_2, then transmit it.
For:	ACK	ACK
	If FL=0 or 2 then do:	If FL=0 or 2 then do:
	$A_1 \leftarrow B_1$	$A_2 \leftarrow B_2$
	$B_1 \leftarrow B_1 + \min(\mu_1, T_c - B_1)$	$B_2 \leftarrow B_2 + \min(\mu_2, T_c - B_2)$
	$FL \leftarrow 0$	$FL \leftarrow 0$
	Go to S_0	Go to S_0
	If FL=1 then do:	If FL=1 then do:
	$B_1 \leftarrow [B_1 - A_1^m(1-p_1)]/p_1$	$A_2 \leftarrow B_2$
	$FL \leftarrow 2$	$B_2 \leftarrow B_2 + \min(\mu_2, T_c - B)$
	Go to S_1	$FL \leftarrow 2$
		Go to S_1

For: CAPT

$A_1 \leftarrow B_1$

Go to S_1

For: NACK

$B_1 \leftarrow B_1 p_1 + A_1(1-p_1)$

Go to S_2

For: CAPT

If FL=2 then use p_2' instead of p_2 until FL is lowered to 0.

Go to S_1

For: NACK

IF FL=0 or 1 then do:

$B_2 \leftarrow B_2 p_2 + A_2(1-p_2)$

Go to S_2

If FL=2 then do:

$B_1 \leftarrow A_1$

Go to S_2

Protocol 3

Nodes of the DG

S_0: If there is a packet with arrival time between A_1 and B_1, then transmit it.

For: LACK, ACK_1

$A_1 \leftarrow B_1$

$B_1 \leftarrow B_1 + \min(\mu_1, T_c - B_1)$

Go to S_0

For: ACK_0

$A_1 \leftarrow B_1$

Go to S_0

Nodes of the NDG

If there is a packet wit arrival time between A_2 and B_2, then transmit it.

For: LACK, ACK_1

$A_2 \leftarrow B_2$

$B_2 \leftarrow B_2 + \min(\mu_2, T_c - B_2)$

Go to S_0

For: ACK_0

Go to S_0

For: NACK

$B_1 \leftarrow B_1 p_1 + A_1(1-p_1)$

Go to S_2

S_1: If there is a packet with arrival time between A_1 and B_1, then transmit it.

For: ACK_1

$A_1 \leftarrow B_1$

$B_1 \leftarrow B_1 + \min(\mu_1, T_c - B_1)$

Go to S_0

For: ACK_0

$A_1 \leftarrow B_1$

Go to S_0

For: NACK

$B_1 \leftarrow B_1 p_1 + A_1(1-p_1)$

Go to S_2

S_2: If there is a packet with arrival time between A_1 and B_1, then transmit it.

For: LACK

$\text{Temp} \leftarrow A_1$

$A_1 \leftarrow B_1$

$B_1 \leftarrow B_1 + (B_1 - \text{Temp})/(1-p_1)$

Go to S_2

For: NACK

$B_2 \leftarrow B_2 p_2 + A_2(1-p_2)$

Go to S_2

If there is a packet with arrival time between A_2 and B_2, then transmit it.

For: ACK_1

$A_2 \leftarrow B_2$

$B_2 \leftarrow B_2 + \min(\mu_2, T_c - B_2)$

Go to S_0

For: ACK_0

Go to S_0

For: NACK

$B_2 = B_2 p_2 + A_2(1-p_2)$

Go to S_2

If there is a packet with arrival time between A_2 and B_2, then transmit it.

For: LACK

$\text{Temp} \leftarrow A_2$

$A_2 \leftarrow B_2$

$B_2 \leftarrow B_2 + (B_2 - \text{Temp})/(1-p_2)$

Go to S_2

For: ACK_1

$Temp \leftarrow A_1$

$A_1 \leftarrow B_1$

$B_1 \leftarrow [B_1 - Temp(1-p_1)]/p_1$

Go to S_1

For: ACK_0

$Temp \leftarrow A_1$

$A_1 \leftarrow B_1$

$B_1 \leftarrow [B_1 - Temp(1-p_1)]/p_1$

Go to S_1

For: NACK

$B_1 \leftarrow B_1\ p_1 + A_1(1-p_1)$

Go to S_2

For: ACK_1

$Temp \leftarrow A_2$

$A_2 \leftarrow B_2$

$B_2 \leftarrow [B_1 - Temp(1-p_1)]/p_2$

Go to S_1

For: ACK_0

$B_2 \leftarrow [B_2 - A_2(1-p_1)]/p_1$

Go to S_1

For: NACK

$B_2 \leftarrow B_2\ p_2 + A_2(1-p_2)$

Go to S_2

Protocol 4

Same as Protocol 3 except for the following changes:

For nodes in the DG:

In state S_0 or S_1:

For: ACK_0

$A_1 \leftarrow B_1$

$B_1 \leftarrow B_1 + \min(\mu_1, T_c - B_1)$

Go to S_0

For nodes in the NDG:

In state S_1:

For: ACK_0

$B_2 \leftarrow A_2 + \min(\mu_2, T_c - A_2)$

Go to S_0

Appendix 2

Let $L^{DG}_{n,k}$, $L^{NDG}_{n,k}$, $L_{n,k}$ and $m_{n,k}$ be as defined in Sec

Let

$$P^1_i(n) = \binom{n}{i} p_1^i (1-p_1)^{n-i}$$

$$P^2_i(n) = \binom{n}{i} p_2^i (1-p_2)^{n-i}$$

Then the following equations govern the evolution of the system:

For Protocol 1:

$$L_{0,0} = 0 \tag{A1}$$

$$L_{1,0} = L_{0,1} = 1 \tag{A2}$$

$$L_{n,0} = P^1_0(n)L_{n,0} + P^1_1(n)(1 + L_{n-1,0}) + \sum_{i=2}^{n} P^1_i(n)L_{i,0} \qquad n \geq 2 \tag{A3}$$

$$L_{0,k} = P^2_0(k)L_{0,k} + P^2_1(k)(1 + L_{0,k-1}) + \sum_{i=2}^{k} P^2_i(k)L_{0,i} \qquad k \geq 2 \tag{A4}$$

$$L_{1,k} = 1 + L_{0,k} \qquad k \geq 1 \tag{A5}$$

$$\begin{aligned} L_{n,k} = {} & P^1_0(n)[P^2_0(k)L_{n,k} + P^2_1(k)(1 + L_{n,k-1}) + \sum_{i=2}^{k} P^2_i(k)L_{0,i}] + \\ & + P^1_1(n)[P^2_0(k)(1 + L_{n-1,k}) + \sum_{i=1}^{k} P^2_i(k)(1 + L_{n-1,0} + L_{0,i})] + \\ & + \sum_{i=2}^{n} \sum_{j=0}^{k} P^1_i(n)P^2_j(k)L_{i,j} \qquad n \geq 2,\ k \geq 1 \end{aligned} \tag{A6}$$

$$L^{DG}_{n,0} = L_{n,0} \qquad n \geq 0 \tag{A7}$$

$$L^{DG}_{0,k} = 0 \qquad k \geq 0 \tag{A8}$$

$$L^{DG}_{1,k} = 1 \qquad k \geq 0 \tag{A9}$$

$$\begin{aligned} L^{DG}_{n,k} = {} & P^1_0(n)[P^2_0(k)L^{DG}_{n,k} + P^2_1(k)(1 + L^{DG}_{n,k-1})] + \\ & + P^1_1(n)[P^2_0(k)(1 + L^{DG}_{n-1,k}) + \sum_{i=1}^{k} P^2_i(k)(1 + L^{DG}_{n-1,0})] + \\ & + \sum_{i=2}^{n} \sum_{j=0}^{k} P^1_i(n)P^2_j(k)L^{DG}_{i,j} \qquad n \geq 2,\ k \geq 1 \end{aligned} \tag{A10}$$

$$L^{NDG}_{n,k} = L_{n,k} - L^{DG}_{n,k} \qquad n \geq 0,\ k \geq 0 \tag{A11}$$

$$m_{0,0} = m_{0,1} = m_{1,0} = 1 \tag{A12}$$

$$m_{n,0} = 1 + P^1_0(n)m_{n,0} + P^1_1(n)(1 + m_{n-1,0}) + \sum_{i=2}^{n} P^1_i(n)m_{i,0} \qquad n \geq 2 \tag{A13}$$

$$m_{0,k} = 1 + P^1_0(k)m_{0,k} + P^2_1(n)(1 + m_{0,k-1}) + \sum_{i=2}^{k} P^2_i(k)m_{0,i} \qquad k \geq 2 \tag{A14}$$

$$m_{n,k} = 1 + m_{0,k} \qquad k \geq 1 \tag{A15}$$

$$\begin{aligned} m_{n,k} = {} & 1 + P^1_0(n)[P^2_0(k)m_{n,k} + P^2_1(k)(1 + m_{n,k-1}) + \sum_{i=2}^{k} P^2_i(k)m_{0,i}] + \\ & + P^1_1(n)[P^2_0(k)(1 + m_{n-1,k}) + \sum_{i=1}^{k} P^2_i(k)(1 + m_{n-1,0} + m_{0,i})] + \\ & + \sum_{i=2}^{n} \sum_{j=0}^{k} P^1_i(n)P^2_j(k)m_{i,j} \qquad n \geq 2,\ k \geq 1 \end{aligned} \tag{A16}$$

Clearly, all the quantities, $L^{DG}_{n,k}$, $L^{NDG}_{n,k}$, $L_{n,k}$, $m_{n,k}$ can be recursively computed from (A1)-(A16).

For Protocol 2:

The same equations (A1)-(A16) govern the system with Protocol 2 except that in equations (A6) and (A16), when n=2, the quantities $L_{n-1,0}$ and $m_{n-1,0}$

should be replaced by $\tilde{L}$ and $\tilde{m}$ that are subsequently calculated.

Let,

$$Q_i(n) = \binom{n}{i} P_2'^{\,i}(1-p_2')^{n-i}$$

Then,

$$\tilde{L}_1 = 1$$

$$\tilde{L}_n = Q_0(n)\tilde{L}_n + Q_1(n)(1 + \tilde{L}_{n-1}) + \sum_{i=2}^{n} Q_i(n)\tilde{L}_i \qquad n \geq 2$$

$$\tilde{m}_1 = 1$$

$$\tilde{m}_n = 1 + Q_0(n)\tilde{m}_n + Q_1(n)(1 + \tilde{m}_{n-1}) + \sum_{i=2}^{n} Q_i(n)\tilde{m}_i \qquad n \geq 2$$

Now,

$$\tilde{L} = 1 + \sum_{i=1}^{\infty} \tilde{L}_i \frac{e^{-x} x^i}{i!}$$

$$\tilde{m} = 1 + \sum_{i=1}^{\infty} \tilde{\ }_i \frac{e^{-x} x^i}{i!}$$

where $x = \lambda_{NDG}\, \mu_2^1$

In the next two protocols we give only the equations for $L_{n,k}$ and $m_{n,k}$. Easily, the equations for $L^{DG}_{n,k}$ and $L^{NDG}_{n,k}$ can be obtained using the same method.

Protocol 3:

$L_{n,k}$ for $n \geq 0$, $k = 0,1$ and $n = 0$, $k \geq 0$ is calculated as in (A1)-(A5) and for $n \geq 2$, $k \geq 1$

$$L_{n,k} = P_0^1(n)[P_0^2(k)L_{n,k} + P_1^2(k)(1 + L_{n,k-1}) + \sum_{i=2}^{k} P_i(k)L_{0,i}] + \\ + P_1^1(n)(1 + L_{n-1,k}) + \sum_{i=2}^{n}\sum_{j=0}^{k} P_i^1(n)P_j^2(k)L_{i,j} \qquad (A17)$$

$m_{n,k}$ for $n = 0,1$, $k \geq 0$ is calculated as in (A12), (A14), (A15). For $n \geq 1$, $k = 0$ we have

$$\tilde{m}_{n,0} = 1 + P_0^1(n)\,\tilde{m}_{n,0} + P_1^1(n)(1 + \tilde{m}_{n-1,0}) + \sum_{i=2}^{n} P_i^1(n)\,\tilde{m}_{i,0} \qquad n \geq 2$$

$$m_{n,0} = 1 + \tilde{m}_{n,0}$$

and for $n \geq 2$, $k \geq 1$, we have:

$$m_{n,k} = 1 + P_0^1(n)[P_0^2(k)m_{n,k} + P_1^2(k)(1 + m_{n,k-1}) + \sum_{i=2}^{k} P_n(k)m_{0,i}] + \\ + P_1^1(k)(1 + m_{n-1,k}) + \sum_{i=2}^{n}\sum_{j=0}^{k} P_i^1(k)P_j^2(k)m_{i,j} \qquad (A18)$$

Protocol 4

$L_{n,k}$ for $n \geq 0$, $k = 0$ and $n = 0$, $k \geq 0$ is calculated as in (A1)-(A4).

$$L_{1,k} = 1 \qquad k \geq 1$$

and for $n \geq 2$, $k \geq 1$ the recursion (A17) is used.

$m_{n,k}$ for $n \geq 0$, $k = 0$ and $n = 0$, $k \geq 0$ is calculated as in (A12)-(A14).

$$m_{1,k} = 1 \qquad k \geq 1$$

and for $n \geq 2$, $k \geq 1$ the recursion (A18) is used.

References

[1] R. G. Gallager, "Conflict Resolution in Random Access Broadcast Network", in Proc. AFOSR Workshop Commun. Theory Appl., Provincetown, MA, Sept. 17-20, 1978, pp. 74-76.

[2] J. Mosely, "An Efficient Contention Resolution Algorithm for Multiple Access Channels", Lab. Inform. Decision Syst., Massachusetts Inst. Technol., Cambridge, MA, Report LIDS-TH-918, June 1979.

[3] J. I. Capetanakis, "Tree Algorithms for Packet Broadcast Channels", IEEE Trans. Inform. Theory, Vol. IT-25, pp. 505-515, Sept. 1979.

[4] I. Cidon and M. Sidi, "The Effect of Capture on Collision Resolution Algorithms", EE Pub. No. 45 , Technion-Israel Institute of Technology, July 1983.

[5] J. J. Metzner, "On Improving Utilization in ALOHA Networks", IEEE Trans. on Communications, Vol. COM-24, pp. 447-448, April 1976.

[6] N. Abramson, "The Throughput of Packet Broadcasting Channels", IEEE on Communications, Vol. COM-25, No. 1, pp. 117-128, Jan. 1977.

Acknowledgement

I would like to thank the Rothschild Foundation for their generous support, Israel Cidon for his very helpful comments and suggestions, and Prof. R. G. Gallager for inviting me to spend the year at M.I.T.